N90-27325

DOCUMENTATION AND KNOWLEDGE ACQUISITION

Daniel Rochowiak — Johnson Research Center
Warren Moseley — Computer Science
University of Alabama in Huntsville

ABSTRACT

Traditional approaches to knowledge acquisition have focused on interviews. An alternative focuses on the documentation associated with a domain. Adopting a documentation approach provides some advantages during familiarization. A Knowledge Management Tool has been constructed to gain these advantages.

INTRODUCTION

The familiarization aspect of knowledge acquisition (KA) continues from the beginning of a knowledge based programming project until the final content for the project is determined. Familiarization, in this sense, is not a distinct episode of knowledge engineering, but consists in all those activities which the knowledge engineer (KE) engages to prepare for the project and to prepare for particular knowledge acquisition sessions. It is important to note that these preparatory activities are both important and time consuming. They are important since they lay the ground for shared common content about the domain, and are time consuming since the knowledge engineer is required to become acquainted with terms, concepts, methods, and theories that may be far different from those with which he or she has already become familiar. In the familiarization process at the beginning of the project the knowledge engineer attempts to find the sources of important information, organize that information, read the documents, charts and other materials that have been assembled, and gain an elementary mastery of the vocabulary of the domain. As the project progresses the KE will continually need to become familiar with new material. However, the familiarization process for knowledge acquisition sessions based on this new material ideally should be less time consuming since a base has been established by previous efforts.

Familiarization is document-driven. Documents play a primary role even when there is a mentor to guide the KE through the material. The documents become a base on which KA can proceed. There are two reasons for this. (9) The first is practical. In order to interact effectively with an expert, the KE and the expert must have some shared conception of the domain. The shared conception is not to be understood as a detailed, precise, accurate or comprehensive account of the domain. Rather the shared account is the base that will continue to develop in the KA process. If the interaction with the expert requires that there be some common understanding, then it should be clear that in the beginning this must be provided in a way other than the interview process. In general, the information needed to establish this shared level of understanding is contained in documents associated with the expert's domain. The second reason is structural. Organizations collect knowledge in documents. These documents represent the stored knowledge of the organization. As such, the knowledge in these documents is social and intersubjective, and constitutes the background against which both individual knowledge and expertise are defined. Thus, for practical and organizational reasons the familiarization aspect of KA is document-driven.

The focus on documentation generates advantages.

- Documents are often "approved" knowledge sources.
- The writers of the documents have "decompiled" to some degree the domain knowledge.
- Documents tie down references in the knowledge dictionary.

- Documents make multiple lines of reasoning available.
- The documents provide a context needed to gain access to specific expertise.
- The documents provide a source of material for both explanation and help facilities.
- Attention to the documentation leads to the identification of weaknesses in the documents.
- Attention to the documents provides a point of reference for the expert and the user.
- Attention to documents leads to tighter coupling and resource-sharing between KE and technical writer.

Methods and aids must be devised to gain these advantages, however.(4) There are at least two ways in which such methods and aids might be built. The first focuses on the direct analysis of existing documents. The objective of this way is to create tools that would directly analyze documents and abstract knowledge. The second way focuses on the management of the familiarization process associated with the documents. We have adopted the latter way. The methods and aids that we are developing focus on the idea of a knowledge dictionary that is similar to the idea of a data dictionary in traditional database operations, and the expanded model of reasoning articulated by Toulmin, Rieke, and Janik (10).

A Documentation Approach to Knowledge Acquisition

Traditional approaches to knowledge engineering emphasize the interview process. Interview driven methods assume that interviewing an expert is the best way to acquire knowledge that is "chunked" and "compiled". Knowledge is "chunked" when items of knowledge are organized into meaningful units. Such chunking is believed to increase the performance of human experts. Unfortunately, such chunking makes knowledge acquisition more difficult especially when such chunks are "compiled"."Compiled" knowledge is knowledge that has been distilled and abstracted of all unnecessary elements; elements which may have originally been needed to gain the knowledge are removed. Further, the organizational schemas may be altered to increase the efficiency of recalling the items in the chunks. Compiled chunks may account for the fact that experts recall all of the content of one chunk before processing a subsequent chunk.

Knowledge engineers are familiar with the problems of chunked and compiled knowledge, and have developed various techniques for acquiring various kinds of knowledge. The documentation approach is consistent with the assumption that knowledge is chunked and compiled, and adds to the available techniques, especially those available during the familiarization activities of knowledge acquisition. Such familiarization activities are part of the episodic units in knowledge acquisition. The episodic units of knowledge acquisition include preparing, conducting, and reviewing interviews. The preparation activities which include familiarization are most intensive during the initial phase of a project. Although it is difficult to obtain data on this topic, informed, but informal, estimates suggest that during the initial phase of a project the ratio of preparation time to session is as high as 8 to 1, while over the life of the project the ratio might be closer to 3.5 to 1. (3) In either case it is clear that a significant portion of a knowledge engineer's time is spent in preparation activities and that such activities are more time consuming during the beginning of the project.

Preparation during the initial phase of a project is a complex undertaking. The knowledge engineer's activities are geared to becoming familiar with the domain. But how does one become familiar with a domain and in what does that familiarity consist? Our suggestion is that the KE becomes familiar with the domain through documents and that this familiarity leads to the production of a knowledge dictionary.

During preparation, the KE attempts to amass documents about the domain. Such documents include text books, reports, instructional materials, design plans, and, in general, any written (hard copy or electronic) materials about the domain. In using the documentation approach, it is assumed that documents have a degree of authority for the experts in the domain, that the experts in the domain would recognize the authority of the documents, and that documents are written and revised in order to establish common understandings and frameworks. We do not assume that there is any direct correspondence between the chunks and terms identified in the documents and those used by domain experts. We do assume, however, that the documents act as constraints on the domain expert. In this sense, the documents constitute an official and authoritative framework within which the expert brings his or her skill to bear. These documents act as a backdrop for two important KEing tasks: building a knowledge dictionary and analyzing it.

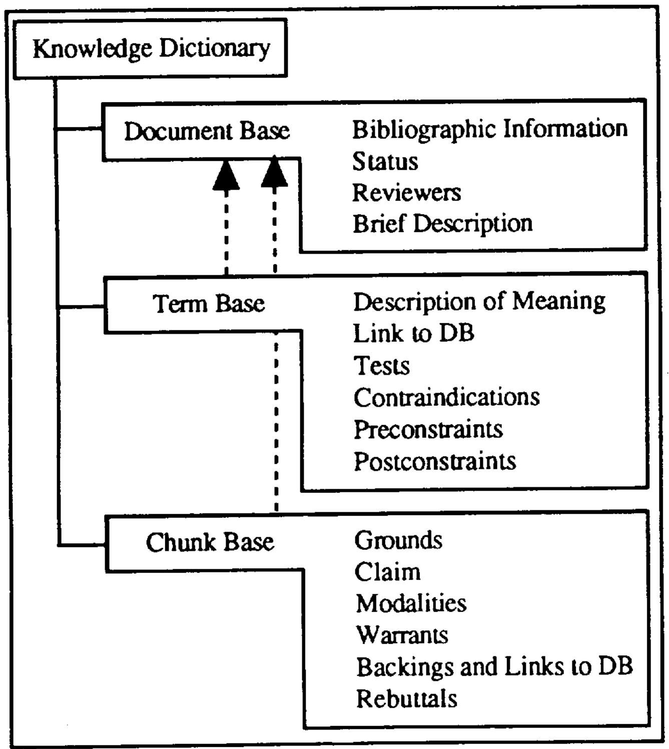


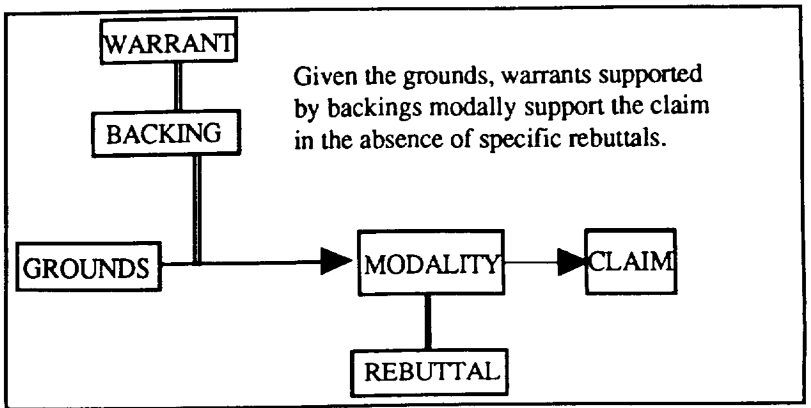
Figure 1

A knowledge dictionary, in its initial formulation, consists of a document base, a term base, and a chunk base. (See Figure 1) These three bases provide a map of the domain and a "first pass" collection of materials for automation.

The document base consists of bibliographic material, the status of the document, indications of whether and by whom a document has been reviewed, and a brief general description of the content and utility of the document. The materials in the term and chunk bases are keyed to these document base. Since in many cases the documents undergo revision as the project evolves, keying the terms and chunks to the document base provides a way of systematically reviewing the materials in the knowledge dictionary in light of revised documentation.

The term base provides information about the meaning and application of the term. An entry for a term provides a brief ordinary language description of the term and any appropriate abbreviation or symbol for it. Additionally, an entry contains typical information about the values the term may take, tests associate with the term, contraindications for the application of the term, and pre and post constraints on the application of the term. The specification of the source for the information provides a link to the documents base.

Knowledge in the chunk base is represented using the Toulmin, Rieke, and Janik (TRJ) model of reasoning. (See Figure 2). Using this model a knowledge chunk is treated as an argumentive or inferential structure, However the model allows for greater depth and flexibility than more strictly logical models. When working with documents, one notices the flexibility of arguments. Even in highly technical areas, assumptions and premises are often not made explicit. (6) Further, the use of


WARRANT
BACKING
GROUNDS
MODALITY
CLAIM
REBUTTAL
Given the grounds, warrants supported by backings modally support the claim in the absence of specific rebuttals.
Figure 2

language makes it possible to link various knowledge elements in subtle, but important ways. The TRJ model more closely represents this sort reasoning.

In the TRJ model a claim is analogous to a conclusion in a deductive model or the the facts that are added to the knowledge base after a rule is fired in a rule based system. The grounds are analogous to premises or the facts in the knowledge base before a rule is fired. The modality can be thought of as a confidence value or some other measure of the strength of the claim based on the grounds. The warrant is most often the conditional statement that allows the grounds to lead to the claim. The warrants may be expressed as a rule, but other representations are possible. The backing indicates the support or basis for the warrant. The rebuttal indicates the considerations that would inhibit or prevent the assertion of the claim.

The TRJ model of reasoning is much more flexible than traditional models that emphasize the logical (propositional and predicate) style of representation. First, it should be noted that a warrant might have multiple backings. If this is so, then the removal of any one of the backings is not sufficient for retracting a warrant. This suggests that a modal logic might be applied to reasonings about warrants. This has been explored in Rochowiak (5). There a very minimal modal system, T, showed promise. Second, the use of the rebuttal notion may prove valuable in nonmonotonic reasonings. For example, if the reasoning unit were implemented in a frame like structure, then the rebuttal slot could be used as a trigger for retracting the rule's application and the retraction of the facts asserted in the claim. Or, it could be used to prohibit the application of a rule that would otherwise match a pattern in the facts. Third, it should be noticed that the notion of a backing provides a very natural way to include references to documents and can be easily extended to include the statements of experts in interview situations. Finally, the availability of backings for warrants (rules) allows for a clearer separation of the system and domain oriented notions of explanation . (7,8).

Given a knowledge dictionary composed of the bases specified above how is it to be analyzed? This question can begin to be given an answer by specifying the sorts of operations that a KE would want to have performed on the bases.

Beginning with the simplest case the KE should be advised of possible alteration sites when a document is updated, revised, deleted, or in some way altered. In an effort to build an essentially bureaucratic system this would be of great importance. A bureaucratic system is one that attempts to automate some process in a bureaucracy. The administration of loans and the purchasing of parts are typical examples. In these cases new rules or forms may require an alteration in either the term or the chunk bases. Another important arena is that in which the KE activity is occurring while the domain is being constructed. In this arena changes to the domain in terms of designs or specifications may force changes in the dictionary. An operation for alerting the KE to potential changes is needed for the analysis of the dictionary. A more complex case involves the grouping and reporting of the materials in the dictionary. Operations that would provide reports of how terms, chunks, and documents are linked, as well as the frequency of such linking, would help the KE to better understand how the knowledge is clustered. At another level operations that would identify some gaps or sites for decomposition are desirable. Such operations might begin with the identification of terms used in the chunks that are not defined or the identification of missing elements in the definition of the term. The identification of empty elements in the chunks would be equally important. Additional operations would be desirable for allowing multiple views of the knowledge dictionary, tracing of particular elements (say particular backings or typical tests) through the knowledge dictionary, and identifying links between chunks (grounds to claims, claims to grounds). Finally in keeping with the spirit of the documentation approach, there should be operations that would generate relevant sections of the knowledge dictionary as a document. Such documents would be useful in creating reports, setting agendas for interviews, and constructing materials for interviews. This is not, of course, an exhaustive account of the sorts of operations that a KE might need in analyzing the knowledge dictionary, but it is indicative of the kinds of concerns that are relevant in the construction of a tool for generating and analyzing a knowledge dictionary.

The process of analyzing and updating the knowledge dictionary is one that will continue over the life of the project. Ideally, the final dictionary would contain all of the information required to document the knowledge aspect of the finished system. For example, the coding that implements a chunk should be tied to the dictionary. This sort of tie would facilitate the identification of the locations in the code that need to be update as a result of some change in the domain knowledge.

A TOOL FOR THE DOCUMENTATION APPROACH

A tool that implements the documentation approach to knowledge acquisition is being developed. "Knowledge Management Tools" (KMT) is constructed primarily in HyperTalk ™ and CLIPS for the Macintosh™ family of computers. The use of a hypertext system is desirable since the hypertext facilities are of themselves useful in KA activities. (1,11). CLIPS is being used since it provides a readily available inference system.

The associational character of the construction and analysis of the knowledge dictionary strongly suggests that a hypertext system is appropriate. During familiarization the entry of data is not strongly structured. The KE may obtain information on one topic and then another without there being a clear connection between the units of information. However, as more information is entered into the dictionary it is reasonable to think that patterns will emerge. These patterns can be quickly and easily captured in associational links. Such links can provide a map of the material in the dictionary and represents the KE's view of the structure of the domain. Further, in familiarization the KE may become aware of new elements that should be added to the dictionary only in the process of reviewing information already entered. This again suggests that an associational link should be created that will allow the KE to easily add the needed information. Finally when there is more than one KE it will often be necessary to review what another KE has done. This review is again an associational link. Each of these reasons suggests the desirability of using a hypertext approach to the management of the familiarization process.

From a management point of view the knowledge dictionary can be treated as a (nonlinear) text. The production of the text should be such that a KE or a member of a KE team can add additional text to an entry during review. This factor means that the text in the knowledge dictionary not only can serve as the background against which a KE formulates interview sessions, but also is a means of communication for members of a KA team. The knowledge dictionary, in this sense, serves as a shared, common background for further knowledge acquisition. These management features again suggest that a hypertext approach is appropriate.

The inclusion of CLIPS in KMT is both an illustrative and cautionary tale. The inclusion of CLIPS was motivated by practical considerations. CLIPS is readily available, and some projects needed to use CLIPS. Further, since reading CLIPS code can be difficult, the direct association of the CLIPS code and the text in the knowledge dictionary would seem desirable. That is, the material in the knowledge dictionary would indicate what a segment of CLIPS code was intended to represent. On the other hand, this approach leads to an effort to coerce the information and knowledge into CLIPS form. This coercion while having some practical advantages leads to difficulties. Most importantly, rather that trying to capture knowledge and information as would seem to be natural, an effort is made to capture knowledge and information in a way that is amenable to CLIPS. It is almost as if an assumption is made that CLIPS is the appropriate tool for the domain. This difficulty is a general one. The problem can be put clearly in the following way: Should the selection of the tool be a determinate of the KA process, or should the KA process be a determinate of the tool? This essay will not attempt to resolve this difficulty, but it should be noted as a serious one.

KMT-CLIPS includes a K-edit stack, a K-dictionary stack, and a K-document stack. Separation of these three stacks adds to the efficiency of the system. The links between the stacks are in the beginning directed toward the K-document. The system as a whole attempts to keep a list of the associations. As the dictionary develops other links are established. Lists of these associations are also kept by the system. Internally, KMT is a collection of associated lists of association links. Access to the text information stored in the system is provided in various ways be accessing these lists.

The key stack is the K-edit stack. Currently this stack contains four screens. Future screens for analysis are planned. The idea of the K-edit stack is to allow the user to enter knowledge based elements and later provide the CLIPS code. However, this is not required and all materials can be entered at one time. Additionally, CLIPS is interactively available. The K-edit stack implements the idea of a chunk base only in a partial way in its rule card. Backings for the warrants are limited to References. Further since CLIPS is a rule based inference system the grounds and claims components of the TRJ model are identified as Conditions and Actions. The idea of a term base is also only partially implemented in the parameter card. The parameter card does not contain fields for all of the features identified above.

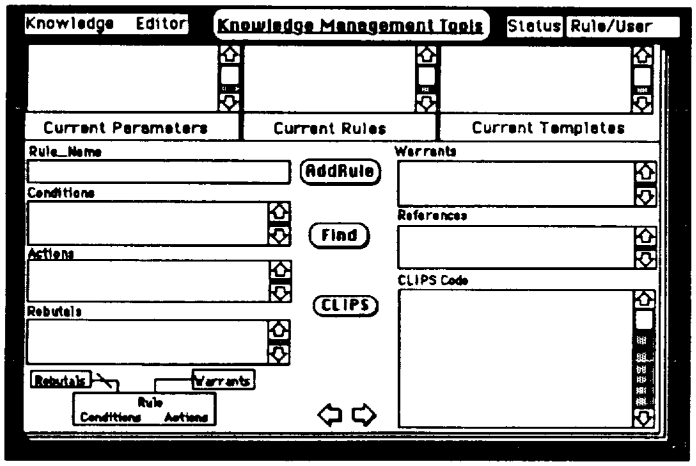


Figure 3

The first card of K-edit is a rule card that contains a chunk template. (See Figure 3) The user provides the name of the rule, its conditions, and actions as ordinary text. Rebuttals are specific circumstances that would prevent the application of the rule. Warrants are the reasons why the rule is being asserted. Both of these are ordinary text, as is the field for references. The diagram in the lower left illustrates the structure. At the top of this card and every card is a list of the currently known parameters, rules, and templates. This provides an interactive access to other parts of the knowledge dictionary. The entries for the term base are found in the Current Parameter and Current Template list. These identifications were selected to provide a more CLIPS-like interface.

The field at the lower right is used for CLIPS code. This may be added or not at the time the chunk is entered. Additionally, it can be tested by clicking the CLIPS button. This button will add the CLIPS code to a user specified text file, and additionally, if desired, load CLIPS and place that file in a buffer. The buffer can then be compiled and run. On exiting CLIPS, the user will be returned to the stack with the clipboard in tact. Thus, if modifications are made to the CLIPS code in the CLIPS environment, those changes can be copied and pasted into the card. It might be handy to have the new rule load into a buffer and then load the previous rules or templates into another buffer. The KE can then paste the new rule into the old CLIPS code and test it. When it is the way the KE wants it, it can be saved as a CLIPS code file. If this approach is taken the KE will need to clean up the dictionary at a latter date. By treating the CLIPS code as a document and building tools that understand CLIPS code, cleaning up the dictionary will be much easier. We are currently developing such tools.

The AddRule button adds the rule elements to a database indexed by the name. The Find button allows the user to find previous elements in the different bases. By selecting an element from those currently known and clicking Find, the user is taken to the database element in K-dictionary. Clicking the Return button in K-dictionary will return to the current card.

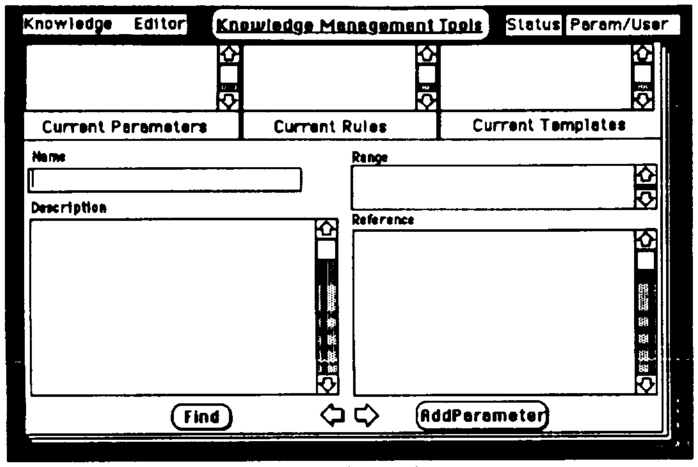


Figure 4

The parameter card partially implements the idea of a term base and functions in a way similar to the rule card. (See Figure 4) The parameters are used to identify the terms in the dictionary. Currently, only the description of the meaning of the term, the range of values, and the reference for the term are included. Fields for additional elements can be added. The AddParameter button adds the data in the fields to the database indexed by name in K-dictionary. Find will find the selected item as in the case of the chunk cards.

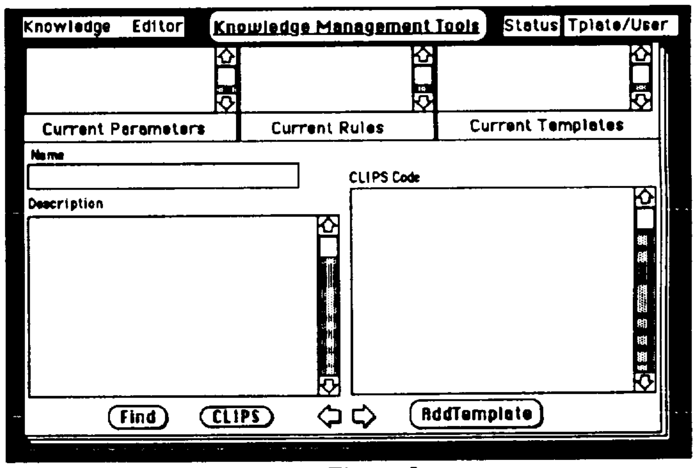


Figure 5

The template card is again similar to the rule card. (See Figure 5) The idea of a template is needed in order to make the general ideas of the documentation approach amenable to the latest version of CLIPS (4.3). A template is a structure that is somewhat similar to a frame and allows for more flexible access to and modification of facts. The template card also provides access to CLIPS since CLIPS code is used to define templates. It should be noted that the file to which the template is saved will load into the CLIPS buffer. If there is any additional editing to be done, it can be done there. The AddTemplate button adds the field to the database in K-dictionary indexed by name. Find will find the selected dictionary item

Currently work is under way to provide more of the features of the documentation approach and to generalize KMT. While there are practical reasons for orienting the system toward a specific inferencing mechanism, that selection also brings problems. The focus on rules and the need for a specific template card are examples. From the beginning, the KE is thinking in terms of the concepts and structures that will ultimately be used in the encoding of the knowledge rather than the knowledge itself. In an ideal case, the software should conform to the knowledge, rather than the knowledge conforming to the software. In improving KMT a more general approach will be taken.

The generalization of KMT will allow for alternative ways of entering information and provide a greater integration with tools that can be used in the interview process. Treating KMT as a "poor man's" knowledge acquisition tool, provides a way of adding different strategies. (4) Of particular interest are the additional representational strategies found in BDM-Kat and MacKat. (2)

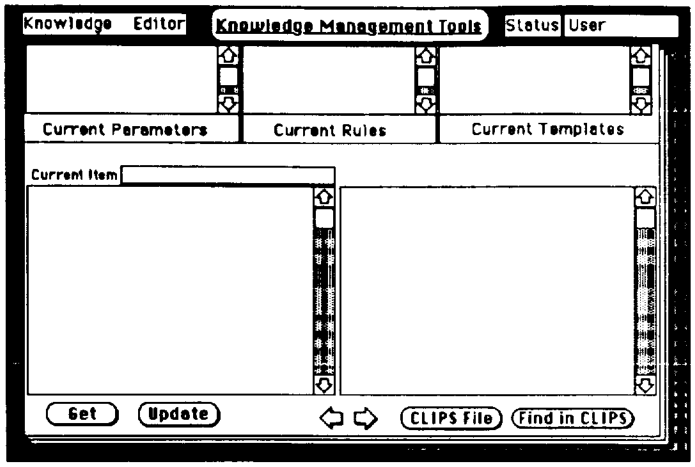


Figure 6

The compare card is used to generate comparisons between the database and the current CLIPS file. The CLIPS File button allows the KE to select a CLIPS knowledge base. The knowledge base is loaded into the field on the right of the card. Selecting a current parameter, rule, or template and clicking the Get button will load the database item from K-dictionary into the field at the left, place its name into the current item field, and move the CLIPS code to the first occurrence of the item. The items text can then be edited. Copy and paste can be used on the two fields. The Update button updates the database in K-dictionary. Similar procedures allow the KE to update the CLIPS code.

While the compare card performs several useful functions, there is much more that it should be able to do. Currently several features are being added that improve on the analysis capabilities of KMT and make greater use of inferencing about the material in the cards. For example, scanning the materials in either of the two fields should be able to produce a list of common items, and a list of items contained in the CLIPS code but not in the database. This would alert the user to check the common elements and to determine if new entries are needed for the elements in the CLIPS code that do not match terms in the dictionary.

K-dictionary and K-document currently share the same structure. The fields on the cards and the operations available are, however, easily tailored to specific needs.

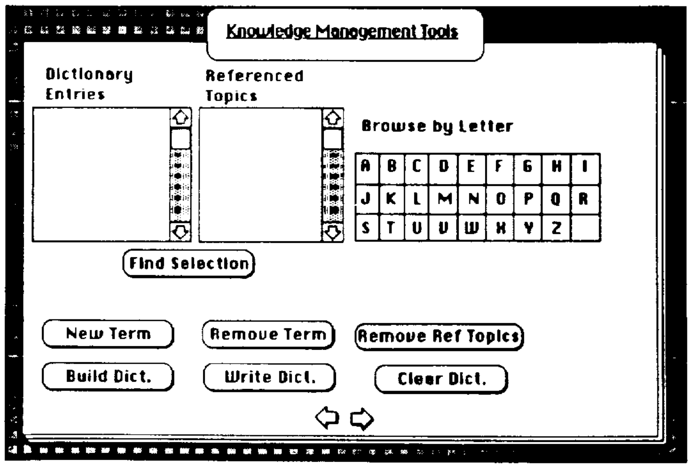


Figure 7

The main card for the two stacks controls the operations of the stack.(See Figure 7) The buttons along the bottom of the card allow the KE to enter or alter the material in the stack rather freely. The New Term Button allows the user to enter a new term or document into the appropriate stack and indexes the entry. The Remove Term button removes the term and updates the index. In each case the scrolling list in the Dictionary Entries field is updated and alphabetized. The Write Dict. button allows the KE to build a text file of the materials in the stack. This file can be imported into a word processor, or saved as a separate file that can be loaded by Build Dict. This allows the user to operate with multiple files. The Clear Dict. button is used in conjunction with the two previous buttons to initialize the stack and prepare for a new file. The Browse by Letter area allows the user to select a letter and browse the entries for that letter. The scrolling list in the Referenced Topics field operates in two ways. In the first way the KE simply create a list of terms that he or she needs to add to the stack. Selecting one of these terms and clicking the Find Selection button will notify the user if the term already exists. If it does not the KE can then enter the information. The Find Selection button also works in connection with the items in the Dictionary Entries field. The Remove Ref Topics button clears the Referenced Topics field. Items in this field can also be cleared manually.

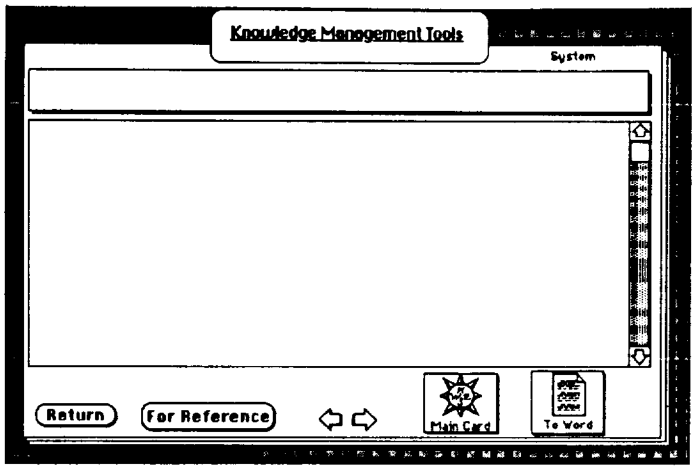


Figure 8

The cards that store the information currently have minimal structure. (See Figure 8) The information can, however, be structured in multiple fields, and resources in the stack allow the information to be gathered in multiple ways. The buttons on the bottom of the card provide a number of functions. The To Word button links to the KE's word processor. The word processor is loaded with the clipboard in tact so that the KE can paste the information into the document. The Main Card button takes the user to the main card of the stack. By highlighting material in the cards entry field and clicking the For Reference button, the Referenced Topic field of the main card is updated. This allows the KE to scan through a stack and quickly note terms that need definitions. The Return button takes the KE back to the K-edit stack, if this stack was entered through it.

The K-dictionary and K-document stacks currently share the same structure and are only distinguished by their content. Links can be established between the two stacks in several ways. We are currently working on ways to make the linking of the information in the three stacks easier. It should also be noted that the K-dictionary and K-document stack contain resources for formulating a frequency-recency model of the user interaction. This may prove to be helpful when an expert is allowed to view the stack or when tracing the flow of knowledge through a stack.

CONCLUSION

The documentation approach to the management of knowledge acquisition provides a way in which the familiarization aspect of knowledge acquisition can be made more productive. The emphasis on existing documentation, especially in bureaucratic systems or systems in the design phase, can be significant. The KMT tools partially implement the documentation approach. The KMT tools have been used on several projects and have been very useful. It should be remembered that the documents in a bureaucracy have been the traditional repository for its knowledge. The documentation approach is directed toward making use of this repository and augments the classical interview approach to knowledge acquisition.

ACKNOWLEDGEMENTS

This research has been supported by Marshall Space Flight Center, NAS8-36955 / D.O. 50, "Space Station Software Automation Research."

REFERENCES

(1) Barrett , Edward(ed). *Text, ConText, and HyperText* (Cambridge, Mass.: MIT Press, 1988).

(2) McGraw, Karen L. "Developing a cognitively-based toolkit for knowledge acquisition," *Workshop on Knowledge Acquisition* (1989) 7-9.

(3) McGraw, Karen L. and Karan Harbison-Briggs. *Knowledge Acquisition* (Englewood Cliffs: Prentice Hall, 1989).

(4) Moseley, Warren. Final Report for FAST-1, "Automated Software Tools," 1989.

(5) Rochowiak, Daniel. "Expertise and reasoning with possibility," *Proceedings of the Second Conference on Artificial Intelligence for Space Applications*, 1987.

(6) ________ "Extensibility and completeness: an essay on scientific reasoning," *The Journal of Speculative Philosophy* 2 (1988): 241-266.

(7) ________ "Simple explanation and reasoning," *Proceedings of the AAAI'88 Workshop on Explanation,* 1988, 95-98.

(8) Rochowiak, Daniel, B. Ragsdale, and L. Wurzelbacher. "An integrated hypertext and rule based system for explanation," *Proceedings of Expert Systems '89*, 1989, 345-352.

(9) Rochowiak, Daniel, D. Hays, and D. Ford. "Document driven knowledge acquisition in the construction of expert systems for aquaculture," *Proceedings of Expert Systems '89*, 1989, 109-120.

(10) Toulmin, S., R. Rieke, and A. Janik, *An Introduction to Reasoning* (New York: Macmillan, 1984).

(11) Wells, Tracy. "Hypertext as a means for knowledge acquisition," *SIGART Newsletter* 108 (April 1989): 136-138.